Complexity of complexity and maximal plain versus prefix-free Kolmogorov complexity

Bruno Bauwens*[†]

March 1, 2012

Abstract

Peter Gacs showed [1] that for every n there exists a bit string x of length n whose plain complexity $C\,(x)$ has almost maximal conditional complexity relative to x, i.e., $C\,(C\,(x)|x) \geq \log n - \log^{(2)} n - O(1)$. Here $\log^2(i) = \log\log i$ etc. Following Elena Kalinina [3], we provide a game-theoretic proof of this result; modifying her argument, we get a better (and tight) bound $\log n - O(1)$. We also show the same bound for prefix-free complexity.

Robert Solovay's showed [10] that infinitely many strings x have maximal plain complexity but not maximal prefix-free complexity (among the strings of the same length); i.e. for some c: $|x| - C\,(x) \leq c$ and $|x| + K\,(|x|) - K\,(x) \geq \log^{(2)} |x| - c\log^{(3)} |x|$. Using the result above, we provide a short proof of Solovay's result. We also generalize it by showing that for some c and for all n there are strings x of length n with $n - C\,(x) \leq c$, and $n + K\,(n) - K\,(x) \geq K\,(K\,(n)|n) - 3\,K\,(\;K\,(K\,(n)|n)\;|n) - c$. This is very close to the upperbound $K\,(K\,(n)|n) + O(1)$ proved by Solovay.

Introduction

Plain Kolmogorov complexity $C\,(x)$ of a binary string x was defined in [4] as the minimal length of a program that computes x. (See the preliminaries or [2, 5, 9] for the details.) It was clear from the beginning (see, e.g., [12]) that complexity function is not computable: no algorithm can compute $C\,(x)$ given x. In [1] a stronger non-uniform version of this result was proven: for every n there exists a string x of length n such that conditional complexity $C\,(C\,(x)|x)$, i.e., the minimal length of a program that maps x to $C\,(x)$, is at least $\log n - O(\log^{(2)} n)$.

*The author is grateful to Elena Kalinina and (Nikolay) Kolia Vereshchagin for giving the text [3]. The author is also grateful to (Alexander) Sasha Shen for his very generous help: for reading earlier texts on these results, for discussion, for providing a clear exposition of section 1 and some parts of section 2, and for his permission to publish it (with small modifications).

[†]Instituto de Telecomunicações Faculdade de Ciência da Universidade do Porto. Supported by the Portuguese science foundation FCT (SFRH/BPD/75129/2010), and is also partially supported by the project CSI^2 (PTDC/EIAC/099951/2008).

(If complexity function were computable, this conditional complexity would be bounded.)

In Section 1 we revisit this classical result and improve it a bit by removing the $\log^{(2)} n$ term. No further improvement is possible because $C\,(n) \leq n + O(1)$, therefore $C\,(C\,(n)|x) \leq \log n + O(1)$ for all x. We use a game technique that was developed by Andrej Muchnik (see [8, 7, 11]) and turned out to be useful in many cases. Recently Elena Kalinina (in her master thesis [3]) used it to provide a proof of Gacs' result. We use a more detailed analysis of essentially the same game to get a better bound.

For some c, a bit string x is C-random if $n - C\,(x) \leq c$. Note that $n + O(1)$ is the smallest upper bound for $C\,(x)$. A variant of plain complexity is prefix-free or self-delimiting complexity, which is defined as the shortest program that produces x on a Turing machine with binary input tape, i.e. without blanc or terminating symbol. (See the preliminaries or [2, 5, 9] for the details.) The smallest upper bound for $K\,(x)$ for strings of length n is $n + K\,(n) + O(1)$. For some c, the string x is defined to be K-random if $n + K\,(n) - K\,(x) \leq c$.

Robert Solovay [10] observed that K-random strings are also C-random strings (for some $c' \leq O(c)$), but not vice versa. Moreover, he showed that some c and infinitely many x satisfy $|x| - C\,(x) \leq c$ and

$$|x| + K\,(|x|) - K\,(x) \geq \log^{(2)} |x| - c \log^{(3)} |x| \,.$$

He also showed that for C-random x the left-hand side of the equation is upper-bounded by $K\,(K\,(n)|n) + O(1)$, which is bounded by $\log^{(2)} n + O(1)$. Later Joseph Miller [6] and Alexander Shen [8] generalized this, by showing that every co-enumerable set (i.e., the complement is enumerable) containing strings of every length, also contains infinitely many x such that the above equation holds. (Note that the set of C-random strings is co-enumerable but the set of K-random strings not.)

In Section 2 we provide a short proof for Solovay's result using the improved version of Gacs' theorem. Then we generalize it by showing that for some c and every n there are strings x of length n with $n - C\,(x) \leq c$ and

$$n + K\,(n) - K\,(x) \geq K\,(K\,(n)|n) - 3\,K\,(\;K\,(K\,(n)|n)\;|n) - c\,.$$

This is very close to the upperbound $K\,(K\,(n)|n) - O(1)$, which was shown by Solovay [10]. By the improved version of Gacs' result, we can choose n such that $K\,(K\,(n)|n) = \log^{(2)} n + O(1)$. For such n we obtain Solovay's theorem with the $c \log^{(3)} |x|$ term replaced by a $O(1)$ constant.

Preliminaries: Let U be a Turing machine. The *plain (Kolmogorov) complexity* relative to U is defined by

$$C_U(x|y) = \min \{|p| : U(p,y) = x\} \,.$$

If the machine U is prefix-free (i.e., for every p, y such that $U(p,y)$ halts, there is no prefix q of p such that $U(q,y)$ halts) then we write $K_U(x|y)$ rather than $C_U(x|y)$, and refer to it as *prefix-free (Kolmogorov) complexity* relative to U.

There exist plain and prefix-free Turing machines U and V for which $C_U(x|y)$ and $K_V(x|y)$ are minimal within an $O(1)$ constant. We fix such machines and omit the indexes U,V. If y is the empty string we use the notation $C(x)$ and $K(x)$.

1 Complexity of complexity can be high

Theorem 1. *There exist some constant c such that for every n there exists a string x of length n such that $C(C(x)|x) \geq \log n - c$.*

To prove this theorem, we first define some game and show a winning strategy for the game. (The connection between the game and the statement that we want to prove will be explained later.)

1.1 The game

Game G_n has parameter n and is played on a rectangular board divided into cells. The board has 2^n columns and n rows numbered $0, 1, \ldots, n-1$ (the bottom row has number 0, the next one has number 1 and so on, the top row has number $n-1$), see Fig. 1.

Initially the board is empty. Two players: White and Black, alternate their moves. At each move, a player can pass or place a pawn (of his color) on the board. The pawn can not be moved or removed afterwards. Also Black may blacken some cell instead. Let us agree that White starts the game (though it does not matter).

The position of the game should satisfy some restrictions; the player who violates these restrictions, loses the game immediately. Formally the game is infinite, but since the number of (non-trivial) moves is a priori bounded, it can be considered as finite, and the winner is determined by the last (limit) position on the board.

Restrictions: (1) each player may put at most 2^i pawns in row i (thus the total number of black and white pawns in a row can be at most $2^i + 2^i$); (2) in each column Black may blacken at most half of the cells.

We say that a white pawn is *dead* if either it is on a blackened cell or has a black pawn in the same column strictly below it.

Winning rule: Black wins if he killed all white pawns, i.e., if each white pawn is dead in the final position.

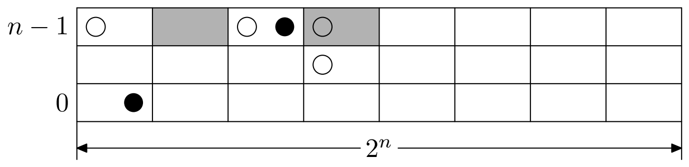

Figure 1: Game board

For example, if the game ends in the position shown at Fig. 1, the restrictions are not violated (there are $3 \leq 2^2$ white pawns in row 2 and $1 \leq 2^1$ white pawn in row 1, as well as $1 \leq 2^2$ black pawn in row 2 and $1 \leq 2^0$ black pawn in row 0). Black loses because the white pawn in the third column is not dead: it has no black pawn below and the cell is not blackened. (There is also one living pawn in the fourth column.)

1.2 How White can win

The strategy is quite simple. White starts by placing a white pawn in an upper row of some column and waits until Black kills it, i.e., blackens the cell or places a black pawn below. In the first case White puts her pawn one row down and waits again. Since Black has no right to make all cells in a column black (at most half may be blackened), at some point he will be forced to place a black pawn below the white pawn in this column. After that White switches to some other column. (The ordering of columns is not important; we may assume that White moves from left to right.)

Note that when White switches to a next column, it may happen that there is a black pawn in this column or some cells are already blackened. If there is already a black pawn, White switches again to the next column; if some cell is blackened, White puts her pawn in the topmost white (non-blackened) cell.

This strategy allows White to win. Indeed, Black cannot place his pawns in all the columns due to the restrictions (the total number of his pawns is $\sum_{i=0}^{n-1} 2^i = 2^n - 1$, which is less than the number of columns). White also cannot violate the restriction for the number of her pawns on some row i: all dead pawns have a black pawns strictly below them, so the number of them on row i is $\sum_{j=0}^{i-1} 2^j = 2^i - 1$, hence White can put an additional pawn.

In fact we may even allow Black to blacken all the cells except one in each column, and White will still win, but this is not needed (and the $n/2$ restriction will be convenient later).

1.3 Proof of Gacs' theorem

Let us show that for each n there exists a string x of length n such that $C\,(C\,(x|n)|x) \geq \log n - O(1)$. Note that here $C\,(x|n)$ is used instead of $C\,(x)$; the difference between these two numbers is $O(\log n)$ since n can be described by $\log n$ bits, so the difference between the complexities of these two numbers is $O(\log\log n)$.

Consider the following strategy for Black (assuming that the columns of the table are indexed by strings of length n):

- Black blackens the cell in column x and row i as soon as he discovers that $C\,(i|x) < \log n - 1$. (The constant 1 guarantees that less than half of the cells will be blackened.) Note that Kolmogorov complexity is an upper semicomputable function, and Black approximates it from above, so more and more cells are blackened.

- Black puts a black pawn in a cell (x, i) when he finds a program of length i that produces x with input n (this implies that $C(x|n) \leq i$). Note that there are at most 2^i programs of length i, so Black does not violate the restriction for the number of pawns on any row i.

Let White play against this strategy (using the strategy described above). Since the strategy is computable, the behavior of White is also computable. One can construct a decompressor V for the strings of length n as follows: each time White puts a pawn in a cell (x, i), a program of length i is assigned to x. By White's restriction, no more than 2^i programs need to be assigned. By universality, a white pawn on cell (x, i) implies that $C(x|n) \leq i + O(1)$. If White's pawn is alive in column x, there is no black pawn below, so $C(x|n) \geq i$, and therefore $C(x|n) = i + O(1)$. Moreover, for a winning pawn, the cell (x, i) is not blackened, so $C(i|x) \geq \log n - 1$. Therefore, $C(C(x|n)|x) \geq \log n - O(1)$.

Remark: the construction also guarantees that $C(x|n) \geq n/2 - O(1)$ for that x. (Here the factor $1/2$ can be replaced by any $\alpha < 1$ if we change the rules of the game accordingly.) Indeed, according to white's strategy, he always plays in the highest non-black cell of some column, and at most half of the cells in a column can be blackened, therefore no white pawns appear in the lower half of the board.

1.4 Modified game and the proof of Theorem 1

Now we need to get rid of the condition n and show that for every n there is some x such that $C(C(x)|x) \geq \log n - O(1)$. Imagine that White and Black play simultaneously all the games G_n. Black blackens the cell (x, i) in game $G_{|x|}$ when he discovers that $C(i|x) < \log n - 1$, as he did before, and puts a black pawn in a cell (x, i) when he discovers an *unconditional* program of length i for x. If Black uses this strategy, he satisfies the stronger restriction: the total number of pawns in row i *on all boards* is bounded by 2^i.

Assume that White uses the described strategy on each board. What can be said about the total number of white pawns in row i? The dead pawns have black pawns strictly below them and hence the total number of them does not exceed $2^i - 1$. On the other hand, there is at most one live white pawn on each board. We know also that in G_n white pawns never appear below row $n/2 - 1$, so the number of live white pawns does not exceed $2i + O(1)$. Therefore we have $O(2^i)$ white pawns on the i-th row in total.

For each n there is a cell (x, i) in G_n where White wins in G_n. Hence, $C(x) < i + O(1)$ (because of property just mentioned and the computability of White's behavior), $C(x) \geq i$ and $C(i|x) \geq \log n - 1$ (by construction of Black's strategies and the winning condition). Theorem 1 is proven.

1.5 Version for prefix complexity

Theorem 2. *There exist some constant c such that for every n there exists a string x of length n such that $C(K(x)|x) \geq \log n - c$ and $K(x) \geq n/2$. This* *also implies that* $K\,(K\,(x)|x) \geq \log n - c$.

The proof of $C\,(K\,(x)|x) \geq \log n - c$ goes in the same way. Black places a pawn in cell (i, x) if some program of length i for a prefix-free (unconditional) machine computes x (and hence $K\,(x) \leq i$); White uses the same strategy as described above. The sum of 2^{-i} for all black pawns is less than 1 (Kraft-inequality); some white pawns are dead, i.e., strictly above black ones, and for each column the sum of 2^{-j} where j is the row number, does not exceed $\sum_{j>i}^{n} 2^{-j} < 2^{-i}$. Hence the corresponding sum for all dead white pawns is less than 1; for the rest the sum is bounded by $\sum_n 2^{-n/2+1}$, so the total sum is bounded by a constant, and we conclude that for x in the winning column the row number is $K\,(x) + O(1)$, and this cell is not blackened.

2 Strings with maximal plain and non-maximal prefix-free complexity

In this section we compare two measures of non-randomness. Let x be a string of length n; we know that $C\,(x) \leq n + O(1)$, and the difference $n - C\,(n)$ measures how "nonrandom" x is. Let us call it C-deficiency of x. On the other hand, $K\,(x) \leq n + K\,(n) + O(1)$, so $n + K\,(n) - K\,(x)$ also measures "nonrandomness" in some other way; we call this quantity K-deficiency of x.

The following proposition means that K-random strings (for which K-deficiency is small; they are also called "Chaitin random") are always C-random (C-deficiency is small; such strings are also called "Kolmogorov random").

Proposition 3 (Solovay [10])**.** $|x| + K(|x|) - K(x) \leq c$ *implies* $|x| - C(x) \leq O(c)$.

Proof. We use a result of Levin: for every string u

$$K\,(u|\;C\,(u)) = C\,(u) + O(1),$$

and, on the other hand, for any positive or negative integer number c:

$$K\,(u|i) = i + c,$$

implies $C\,(u) = i + O(c)$[1].

Let $n = |x|$. Notice that

$$n + K(n) \leq K(x) - c = K(x, n) - O(c) \leq K(x|n) + K(n) - O(c)\,.$$

Hence, $K(x|n) \geq n - O(c)$, thus $K(x|n) = n + O(c)$ and thus: $C(x) = n + O(c)$. □

[1]Textbooks like [5, Lemma 3.1.1] mention only the first statement. To show the second, note that the function $i \mapsto K\,(x|i)$ maps numbers at distance c to numbers at distance $O(\log c)$, hence, the fixed point $C\,(x)$ must be unique within an $O(1)$ constant. Furthermore, for any i, the fixed point must be within distance $O(\log |i - K\,(u|i)|)$ from i, hence $|\,C\,(u) - i| \leq O(\log |i - K\,(u|i)|) = O(\log c)$.

R. Solovay showed that the reverse statement is not always true: a C-random string may be not K-random. However, as the following result shows, the K-deficiency still can be bounded for C-random strings:

Proposition 4 (Solovay [10])**.** *For any x of length n the inequality $C(x) \geq n-c$ implies:*

$$n + K(n) - K(x) \leq K(K(n)|n) + O(c)\,.$$

Note that $K(K(n)|n) \leq \log^{(2)} n + O(1)$.

Proof. The proof uses another result of Levin [1, 2, 5]: for all u, v we have the additivity property

$$K(u,v) = K(u) + K(v|u, K(u)) + O(1)\,.$$

To prove Proposition 4, notice that $n = C(x) = K(x|\,C(x)) = K(x|n)$ with $O(c)$-precision. By additivity we have: $K(x) = K(n,x) = K(n) + K(x|n, K(n))$. Putting these observations together, we get (with $O(c)$-precision)

$$\begin{aligned} n + K(n) - K(x) &= K(x|n) + K(n) - (K(n) + K(x|n, K(n))) = \\ &= K(x|n) - K(x|n, K(n))\,. \end{aligned} \tag{1}$$

Observe that $K(x|n) \leq K(x|n, K(n)) + K(K(n)|n) + O(1)$, hence the K-deficiency is bounded by $K(K(n)|n) + O(c)$. □

The following theorem shows that for all n the bound $K(K(n)|n)$ for K-deficiency for C-random strings can almost be achieved. The error is at most $O(\log K(K(n)|n))$.

Theorem 5. *For some c and all n there are strings x of length n such that $n - C(x) \leq c$, and*

$$n + K(n) - K(x) \geq K(K(n)|n) - 3\,K(\;K(K(n)|n)\;|n) - c\,.$$

By corollary, infinitely many C-random strings have K-deficiency $\log^{(2)}|x| + O(1)$. Indeed, for n such that $K(K(n)|n) = \log^{(2)} n + O(1)$, we have $K(\;K(K(n)|n)\;|n) \leq O(1)$, and hence, a slightly stronger statement than proved by Solovay [10] is obtained.

Corollary 6. *There exists a constant c and infinitely many x such that $|x| - C(x) \leq c$ and $|x| + K(|x|) - K(x) \geq \log^{(2)}|x| - c$.*

Before proving Theorem 5, we prove the corollary directly.

Proof. First we choose n, the length of string x. It is chosen in such a way that $K(K(n)|n) = \log^{(2)} n + O(1)$ and $K(n) \geq (\log n)/2$ (Theorem 2). (So the bound of Proposition 4 is not an obstacle.) We know already (see equation 1) that for a string x with C-deficiency c the value of K-deficiency is $O(c)$-close to $K(x|n) - K(x|n, K(n))$. This means that adding $K(n)$ in the condition should decrease the complexity, so let us include $K(n)$ in x somehow. We also have to guarantee maximal C-complexity of x. This motivates the following choice:

- choose r of length $n - \log^{(2)} n$ such that $K\left(r|n, K\left(n\right)\right) \geq |r|$. Note that this implies $K\left(r|n, K\left(n\right)\right) = |r| + O(1)$, since the length of r is determined by the condition;
- let $x = \langle K(n)\rangle r$, the concatenation of $K(n)$ (in binary) with r. Note that $\langle K(n)\rangle$ has at most $\log^{(2)} n + O(1)$ bits for every n, and by choice of n has at least $\log^{(2)} n - 1$ bits, hence $|x| = n + O(1)$.

As we have seen (looking at equation (1)), it is enough to show that $K\left(x|\, K\left(n\right), n\right) \leq n - \log^{(2)} n$ and $K\left(x|n\right) \geq n$ (the latter equality implies $C\left(x\right) = n$); all the equalities here and below are up to $O(1)$ additive term.

- Knowing n, we can split x in two parts $\langle K\left(n\right)\rangle$ and r. Hence, $K\left(x|\, K\left(n\right), n\right) = K\left(K\left(n\right), r|n, K\left(n\right)\right)$, and this equals $K\left(r|n, K\left(n\right)\right)$, i.e., $n - \log^{(2)} n$ by choice of r.
- To compute $K\left(x|n\right)$, we use additivity:

  $$K\left(x|n\right) = K\left(K\left(n\right), r|n\right) = K\left(K\left(n\right)|n\right) + K\left(r|\, K\left(n\right), K\left(K\left(n\right)|n\right), n\right).$$

  By choice of n, we have $K\left(K\left(n\right)|n\right) = \log^{(2)} n$, and the last term simplifies to $K\left(r|\, K\left(n\right), \log^{(2)} n, n\right)$, and this equals $K\left(r|\, K\left(n\right), n\right) = n - \log^{(2)} n$ by choice of r. Hence $K\left(x|n\right) = \log^{(2)} n + (n - \log^{(2)} n) = n$.

□

Remark 1: One can also ask how many strings exist that satisfy the conditions of Corollary 6. By Proposition 4, the length n of such a string must satisfy $K\left(K\left(n\right)|n\right) \geq \log^{(2)} n - O(1)$. By Theorem 2, there is at least one such an n for every length of n in binary. Hence such n, can be found within exponential intervals.

Remark 2: One can ask for these n, how many strings x of length n satisfy the conditions of Corollary 6. By a theorem of Chaitin [5], there are at least $O(2^{n-k})$ strings with K-deficiency k, hence we can have at most $O(2^{n-\log^{(2)} n})$ such strings. It turns out that indeed at least a fraction $1/O(1)$ of them satisfy the conditions of Corollary 6. To show this, note that in the proof Theorem 5, every different r of length $n - |q| = |n| - \log^{(2)} n + O(1)$ leads to the construction of a different x. For such r we essentially need $K\left(r|n, K\left(n\right), q\right) \geq |r| - O(1)$, and hence there are $O(2^{n-\log^{(2)} n})$ of them.

Proof of Theorem 5. In the proof above, in order to obtain a large value $K\left(x|n\right) - K\left(x|n, K\left(n\right)\right)$, we incorporated $K\left(n\right)$ in a direct way (as $\langle K\left(n\right)\rangle$) in x. To show that $C(x) = K(x|n) + O(1)$ is large we essentially used that the length of $\langle K\left(n\right)\rangle$ equals $K\left(K\left(n\right)|n\right) + O(1)$. For general n, this trick does not work anymore, but we can use a shortest program for $K\left(n\right)$ given n (on a plain machine). For every n we can construct x as follows:

- let q be a shortest program that computes $K(n)$ from n on a *plain* machine (if there are several shortest programs, we choose the one with shortest running time). Note that $|q| = C(K(n)|n) + O(1) = C(q|n) + O(1)$ (remind that by adding some fixed instructions, a program can print itself, and that a shortest program is always incompressible, thus up to $O(1)$ constants: $|q| \geq C(K(n)|n) \geq C(q|n) \geq |q|$), by Levin's result (conditional version), the last term also equals $K(q|n, |q|) + O(1)$;
- let r have length $n - |q|$, such that $K(r|n, K(n), q) \geq |r|$. Note that this implies $K(r|n, K(n), q) = |r| + O(1)$, (since the length of r is determined by the condition).
- We define x as the concatenation qr.

We show that $C(x) = n + O(1)$ and that the K-deficiency is at least $|q| - K(|q|\,|n) + O(1)$. To show that this implies the theorem, we need that

$$K(K(n)|n) - 3K(\,K(K(n)|n)\,|n) \leq C(K(n)|n) - K(\,C(K(n)|n)\,|n) + O(1)\,,$$

which is for $a = K(n)$ the conditioned version of Lemma 7:

$$K(a|n) - 3K(\,K(a|n)\,|n) \leq C(a|n) - K(\,C(a|n)\,|n) + O(1)\,.$$

Following the same structure as the proof above, it remains to show that $K(x|\,K(n), n) \leq n - |q| + K(|q|\,|n)$ and $K(x|n) \geq n$ (the latter equality implies $C(x) = n$); all the equalities here and below are up to $O(1)$ additive term.

- Knowing $|q|$, we can split x in two parts q and r. Hence, $K(x|\,K(n), n, |q|) = K(q, r|n, K(n), |q|)$. Given $n, K(n), |q|$ we can search for a program of length $|q|$ that on input n outputs $K(n)$; the one with shortest computation time is q. Hence, $K(q, r|n, K(n), |q|) = K(r|n, K(n), |q|)$, i.e., $n - |q|$ by choice of r, and therefore $K(x|\,K(n), n) \leq n - |q| + K(|q|\,|n)$.
- To compute $K(x|n)$, we use additivity:

  $$K(x|n) \geq K(x|n, |q|) = K(q, r|n, |q|) = K(q|n, |q|) + K(r|q, K(q|n, |q|), n)\,.$$

  By choice of q we have $C(q|n) = |q|$, and hence by Levin's result $K(q|n, |q|) = |q|$. The last term is $K(r|q, |q|, n)$ which equals $K(r|q, n) = n - |q|$ by choice of r. Hence, $K(x|n) \geq |q| + (n - |q|) = n$. □

Lemma 7. $K(a) - 3K(K(a)) \leq C(a) - K(C(a)) + O(1)$

Proof. Note that $K(a) - C(a) \leq K(C(a))$. Indeed, any program for a plain machine can be considered as a program for a prefix-free machine conditional to it's length. Hence, we can transform a plain program p to a prefix-free program by adding a description of $|p|$ of length $K(|p|)$ to p. Hence it remains to show $2K(C(a)) \leq 3K(K(a)) + O(1)$. Solovay [10] showed that

$$K(a) - C(a) = K(K(a)) + O(K(K(K(a))))\,,$$

hence,

$$|\,K(K(a)) - K(C(a))| \leq O(\log K(K(a)))\,.$$

□

References

[1] P. Gács. *On the symmetry of algorithmic information. Soviet Math. Dokl.*, **15**(5):1477–1480 (1974).

[2] P. Gács. *Lecture notes on descriptional complexity and randomness.* `http://www.cs.bu.edu/faculty/gacs/papers/ait-notes.pdf`, 1988-2011.

[3] E. Kalinina. *Some applications of the method of games in Kolmogorov complexity.* Master thesis. Moscow State University, 2011.

[4] A.N. Kolmogorov, *Three approaches to the quantitative definition of information, Problemy peredachi Informatsii*, vol. 1, no. 1, pp. 3–11 (1965)

[5] M. Li and P.M.B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications.* Springer-Verlag, New York, 2008.

[6] J.S. Miller, *Contrasting plain and prefix-free complexities.* Preprint available at `http://www.math.wisc.edu/~jmiller/downloads.html`.

[7] A. Muchnik, *On the basic structures of the descriptive theory of algorithms*, *Soviet Math. Dokl.*, **32**, p. 671–674 (1985).

[8] An.A. Muchnik, I. Mezhirov, A. Shen, N. Vereshchagin, *Game interpretation of Kolmogorov complexity* (2010), `arxiv:1003.4712v1`

[9] A. Shen, *Algorithmic Information theory and Kolmogorov complexity.* Technical report TR2000-034, Uppsala University (2000).

[10] R.Solovay, *Draft of a paper (or series of papers) on Chaitin's work.* Unpublished notes, 215 pages, (1975).

[11] N. Vereshchagin, Kolmogorov complexity and Games, *Bulletin of the European Association for Theoretical Computer Science*, **94**, Feb. 2008, p. 51–83.

[12] A.K. Zvonkin, L.A. Levin, *The complexity of finite objects and the development of the concepts of information and randomness by means of the theory of algorithms*, *Russian Math. Surveys*, **25**, issue 6(156):83–124, 1970.